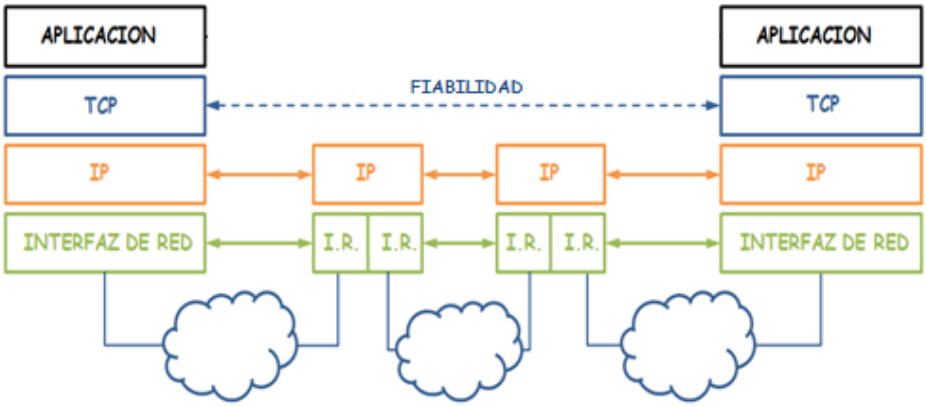
TEMA 5: PROTOCOLOS TCP Y UDP

1. PROTOCOLO TCP



TCP es un protocolo de nivel de transporte que proporciona un servicio de flujo de octetos, orientado a la conexión y, por tanto, fiable y sin congestiones. Además ofrece un servicio multiplexado y dúplex.

- **TRANSFERENCIA DE FLUJO DE OCTETOS (byte-stream).** La entidad TCP emisora recoge, numera, almacena y agrupa los octetos de datos o carga útil en segmentos calculando el MSS (Maximum Segment Size) para que los datagramas IP se correspondan con la MTU de la red de acceso.
- **ORIENTADO A CONEXION.** Mediante la conexión entre la correspondiente pareja de sockets, la entidad TCP emisora se pone de acuerdo con la entidad TCP receptora para llevar a cabo todas las funciones de control de errores y flujo en la transferencia de datos.

  -CONTROL DE ERRORES (Fiabilidad): Abarca tanto los errores lógicos como físicos.

  * **ERRORES LÓGICOS (Octetos de datos perdidos, desordenados y duplicados):**

    Se controlan usando **TEMPORIZADORES**, donde el campo de datos de cada segmento de información tiene asociado un temporizador y as u vencimiento se retransmite el segmento.

    Otra opción es usar **NÚMEROS DE SECUENCIA** asignados a cada octeto transmitido. Cada octeto tiene su propio número de secuencia.

    Por último se pueden utilizar **CONFIRMACIONES** que permiten a la entidad TCP emisora girar su ventana de transmisión para seguir enviando datos. Se asocia al campo de datos del segmento.

  * **ERRORES FISICOS:**

    Consiste en la **DETECCIÓN**, mediante un mecanismo de suma comprobación que se aplica a todo el segmento, y su **CORRECCIÓN** mediante retransmisiones al vencimiento de los correspondientes temporizadores.

  -CONTROL DE FLUJO: Impide que una entidad TCP emisora transmita más rápidamente de lo que otra entidad TCP receptora es capaz de almacenar y procesar.

  El control de flujo lo ejerce la entidad TCP receptora a través de su W_R para evitar que la entidad TCP emisora desborde el buffer de dicha entidad TCP receptora.

- **CONTROL DE LA CONGESTION:** Mecanismo de ventana ejercido por el emisor sobre sí mismo para evitar seguir desbordando el buffer de recepción IP de un router.

  El control de la congestión lo ejerce la entidad TCP emisora a través de su ventana de congestión.

  El control de congestión permite que la entidad TCP emisora baje el ritmo de transmisión cuando ocurre un desbordamiento en el buffer de un router.

- **MULTIPLEXACION:** TCP es multiplexado, es decir, puede dar servicio a varios procesos de aplicación simultáneamente, gracias a los números de puerto que los identifican.
- **FULL-DUPLEX:** La transmisión de información es bidireccional y simultanea.

1.1 VENTANAS DESLIZANTES.

Toda entidad **TCP EMISORA** dispone de una ventana de transmisión (W_T) y de una ventana de congestión (W_C).
Toda entidad **TCP RECEPTORA** dispone de una ventana de recepción (W_R)

- **VENTANA DE RECEPCION (W_R):**

  Garantiza que no se inunde al receptor, ejerciendo un control de flujo sobre el emisor.

  Controla la numeración de los octetos de datos que en un momento dado el receptor puede aceptar en función del tamaño de su buffer de recepción

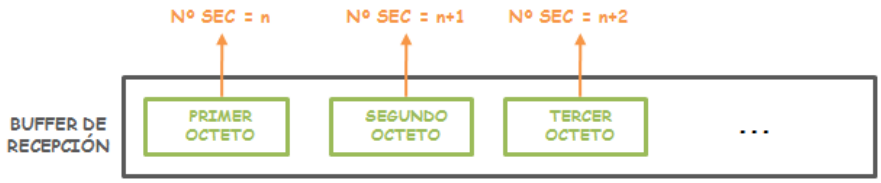
  El valor de WR se calcula a partir del número de octetos libres que se pueden almacenar en el buffer de recepción de la entidad TCP receptora.

  -BUFFER DE RECEPCIÓN:

  Octetos de datos procedentes de la entidad TCP emisora se recogen, numeran y almacenan en el buffer de recepción.

  El control de los números del buffer se realiza mediante la W_R. Los números de secuencia consecutivos de los octetos de datos que en un momento dado el receptor puede aceptar.

  Cuando se llena el buffer, se entregan los datos al proceso de aplicación.



- **VENTANA DE CONGESTION (W_C):**

  Garantiza que no se siga inundando a un router, ejerciendo un control de transmisión en el emisor.

  Indica el número de segmentos enviado en función de la congestión en Internet y se calcula a partir del número de retransmisiones realizadas (por timeouts o pérdida de segmentos de datos y/o confirmaciones)

  Actúa simultáneamente y en paralelo a W_R.

- **VENTANA DE TRANSMISION (W_T):**

  Controla la numeración de los octetos de datos (enviados y no confirmados) almacenados en el buffer de transmisión.

  A medida que van llegando las confirmaciones, se desactivan los temporizadores asociados y se va liberando espacio en el buffer de transmisión para dejar sitio a nuevas copias de otros octetos de datos que se desean transmitir.

  El tamaño de W_T viene determinado por la capacidad del receptor y la capacidad de la red. Viene determinado por el espacio disponible en el buffer de recepción de la entidad TCP receptora y por el espacio disponible en los buffers de recepción IP de los routers intermedios entre el origen y el destino.

$$W_T = \text{Min}\ (W_R,\ W_C)$$

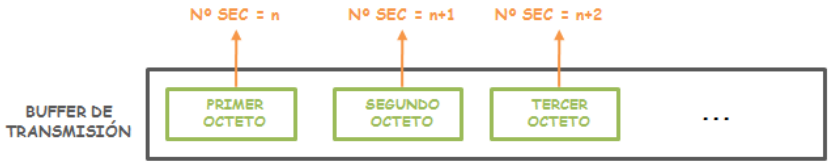
  En cada momento, el emisor tomara en consideración la más pequeña de las dos ventanas para asegurarse de que no desborda al receptor ni provoca, o contribuye a agravar, una situación de congestión en la red.

  -BUFFER DE TRANSMISION:

  Octetos de datos procedentes del proceso de aplicación se recogen, numeran y almacenan en el buffer de recepción.

  El control de los números del buffer se realiza mediante la W_T. Los números de secuencia consecutivos de los octetos de datos que en un momento dado el emisor ha enviado sin haber recibido confirmación.

  Cuando se llena el buffer, se van tomando trozos agrupándolos por segmentos y añadiéndoles una cabecera.



- **SINCRONIZACION DE WR Y W_T PARA CONTROL DE FLUJO:**

> **W_R inicial = TAMAÑO MAXIMO DEL PROPIO BUFFER DE RECEPCIÓN.**

En fase de transferencia de datos, WR va variando puntualmente en función de los octetos libres de su buffer de recepción.

Se pasan datos al proceso de aplicación cuando se llena el buffer de recepción, salvo que el bit PSH=1.

El límite inferior de WR será el primer octeto que se espera recibir después del último octeto pasado al proceso de aplicación.

Se gira WR (límites inferior y superior) cuando se pasan datos al proceso de aplicación (al llenarse el buffer de recepción).

> **W_T inicial = W_R inicial DEL OTRO EXTREMO (TAMAÑO MAXIMO DEL PROPIO BUFFER DE RECEPCIÓN DEL OTRO EXTREMO)**

W_T va variando $W_T \leq W_R$ puntualmente en fase de transferencia de datos en función de la W_R del otro extremo.

Se gira W_T (limites inferior y superior) cuando se recibe confirmación de todos los octetos comprendidos entre el límite inferior y superior de W_T.

1.2 Segmento Tcp.

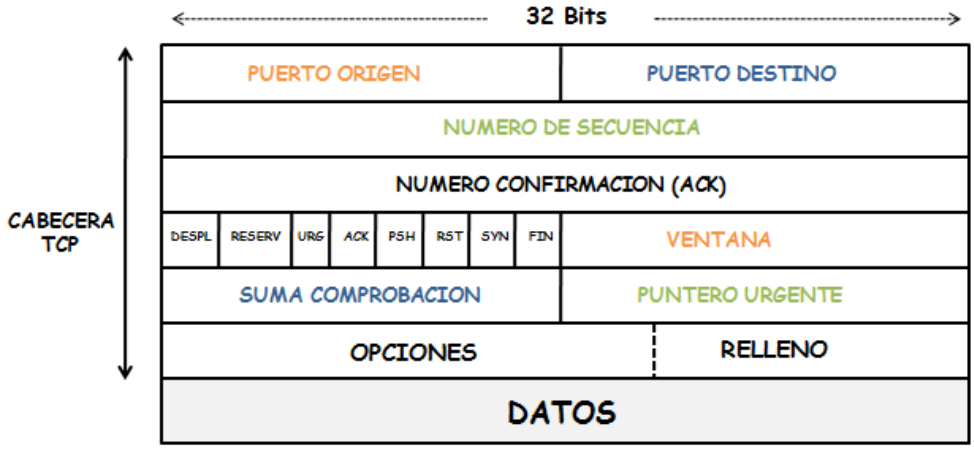
- **DATOS:** Incluye la cabecera de información del nivel de aplicación y los datos del mensaje (si existen).

  - **MSS:** Tamaño establecido con anterioridad a la transferencia que indica el máximo número de bytes que se pueden enviar en un segmento.
    Se calcula de tal forma que los datagramas IP que se vayan a enviar coincidan con la MTU de la red.

> **MSS (Maximum Segment Size) = Carga Útil = SDU del Nivel de Aplicación**

  - **DE OCTETOS A SEGMENTOS:**
    Es necesario que TCP disponga de un temporizador que le permita recoger una cantidad razonable de octetos de datos antes de crear el oportuno segmento de datos.
    1. Primero se guarda una copia de los datos que se quieren transmitir por si hay problemas en el envío y se necesita retransmitir.
    2. Una vez que la entidad emisora ha llenado el buffer de transmisión, toma un trozo de los datos almacenados y les añade una cabecera para generar el segmento.

- **CABECERA:** Tiene longitud mínima, por omisión, de 20 octetos y máxima de 60 octetos incluyendo las opciones.



* **PUERTO ORIGEN/DESTINO:** Limitados a 16 bits cada uno. Como máximo 65536 puertos en un equipo. Identifican al proceso de aplicación emisor / receptor que envía un segmento TCP.
* **NUMERO DE SECUENCIA:** Indica el primer octeto del campo de datos del segmento que se envía.
  No se empieza a contar desde 0, sino desde un número aleatorio (por seguridad).
  Como el numero de secuencia es de 32 bits, se va a trabajar en modulo 2^{32}, por tanto los números de secuencia se numeran cíclicamente del 0 al 2^{32}-1.
* **NUMERO DE CONFIRMACION:** Indica el primer octeto del campo de datos del siguiente segmento que se espera recibir
  Confirma todos los segmentos hasta el número de secuencia recibido.
* **DESP:** Indica el numero de bloques de cuatro octetos que ocupa la cabecera. Como mínimo 20 octetos y como máximo 60 octetos.
* **RESERVADO:** Reservados para un uso futuro.
* **URG:** Si esta activo indica que el campo de puntero urgente es un campo valido y significativo y que por tanto la entidad TCP receptora lo debe analizar debidamente.
* **PSH:** Servicio forzado de transferencia, es decir, se ha tenido que enviar el contenido del buffer de transferencia sin que se haya llenado y por tanto la entidad receptora tampoco esperara a que se llene el suyo.
* **RST:** Bit de reinicio. Si esta activo indica al modulo TCP receptor que abandone la comunicación debido a un error o a una situación anormal.

Se puede usar como respuesta para rechazar una solicitud de establecimiento de una conexión TCP.

* **FIN:** Si esta activo indica que se está solicitando finalización o liberación del enlace.
* **ACK:** Indica que el campo de numero de confirmación es un campo significativo.
* **SYN:** Si esta activo indica que se está estableciendo o procesando una conexión TCP. Si se combina con el bit ACK proporciona los segmentos específicos de control TCP.
  1. ACK=1 + SYN=1 ⟺ Se acepta la solicitud del establecimiento de la conexión.
  2. ACK=1 + SYN=0 ⟺ Solicitud de establecimiento de conexión TCP.
* **VENTANA:** Se utiliza como control de flujo. Indica el número de octetos pendientes de confirmación que el receptor puede manejar en función del tamaño puntual de su buffer de recepción, es decir, el número de bytes que puede transmitir sin necesidad de esperar confirmación.
  El tamaño máximo de la ventana será Tam_Max = 2^{16}-1.

1.3 Opciones Tcp.

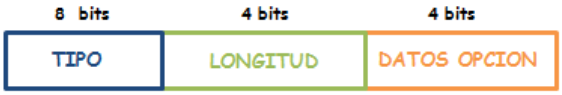
Campo de longitud variable de hasta 40 octetos.

Se van incorporando nuevas opciones poco a poco en las diferentes implementaciones de TCP.

Se especifican en la fase de establecimiento de la conexión, es decir, sólo en aquellos segmentos de control con el bit SYN=1

- **FORMATO:**

  El campo opciones sigue el formato TLV: Tipo-Longitud-Valor (o Datos)



  * **TIPO (8 bits):** Indica, mediante un número, el tipo de opción del segmento TCP.

  * **LONGITUD (8 bits):** Indica la longitud completa de la opción en octetos. (TIPO + LONGITUD+ DATOS)

- **TAMAÑO MAXIMO DE SEGMENTO (MSS):** Indica el número de octetos de la carga útil (campo de datos) que el receptor desea recibir para un procesado más óptimo de dicha carga.

  No es el tamaño del buffer de recepción.

  * **TIPO** = 2 * **LONGITUD**= 4 octetos * **DATOS** = MSS

- **FACTOR DE ESCALA DE VENTANA:** Permite ampliar el campo VENTANA.

  Equivale a usar un máximo de ventana de 32 bits. Por omisión el tamaño máximo de la ventana de recepción es de 2^{16}-1 octetos, pero con esta opción de puede escalar hasta 2^{32}-1 (1GB aproximadamente).

  * **TIPO** = 3 * **LONGITUD**= 3 octetos * **DATOS** = Tamaño de ventana

- **CONFIRMACION SELECTIVA (SACK):** Permite informar al emisor de segmentos de información no contiguos que han sido recibidos correctamente y evitar retransmisiones innecesarias. Implícitamente, indica que octetos de datos se han perdido o no han llegado todavía.

- **MARCA DE TIEMPO (TIMESTAMP):** Permite al emisor calcular el valor RTT (Round Trip Time: tiempo de ida y vuelta de un segmento) para configurar sus temporizadores.

  Se usa en la fase de establecimiento de la conexión, siempre y cuando haya recibido respuesta con SYN= 1, ACK =1.

  * **TIPO** = 8 * **LONGITUD**= 10 octetos

  * **DATOS** = Valor actual reloj emisor (4 octetos) + respuesta eco (4 octetos).

  El valor actual del reloj del emisor es un valor monótonamente creciente, el eco es válido si el bit ACK = 1 en la cabecera del segmento

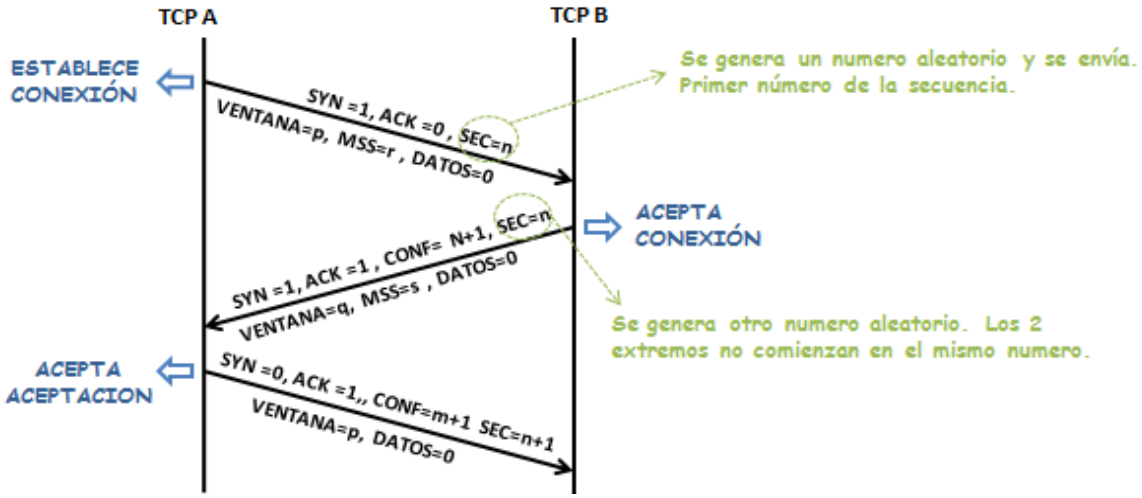
  El receptor devuelve el mismo valor (eco) en el ACK del segmento.

  Protege de errores de duplicación del NUMERO DE SECUENCIA en conexiones de alta velocidad y, por tanto, permite reconocer octetos distintos con el mismo número de secuencia en la misma conexión cuando los números de secuencia dan la vuelta durante el tiempo de vida de dicha conexión.

1.4 Funcionamiento del Protocolo Tcp.(20-25)

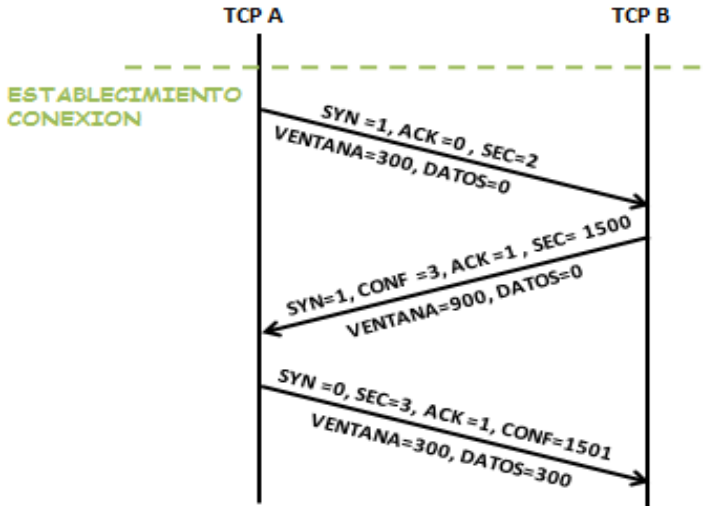
- **ESTABLECIMIENTO DE CONEXION:**

Para establecer una conexión, se necesitan enviar tres segmentos de control en total.



Los números de secuencia no se reutilizan durante un tiempo prudencial para evitar que los octetos de datos de los segmentos de información de una conexión se confundan con los de otra, sobre todo cuando se cierran y abren conexiones inmediatamente.
Los números de secuencia son aleatorios porque si no podrían ocurrir que segmentos obsoletos no se distingan de los actuales ya que los segmentos se pueden perder, retrasarse o duplicarse.
Se va a establecer el tamaño máximo de la ventana de recepción y opcionalmente el tamaño del campo de datos (MSS) de los segmentos que se esperan recibir.

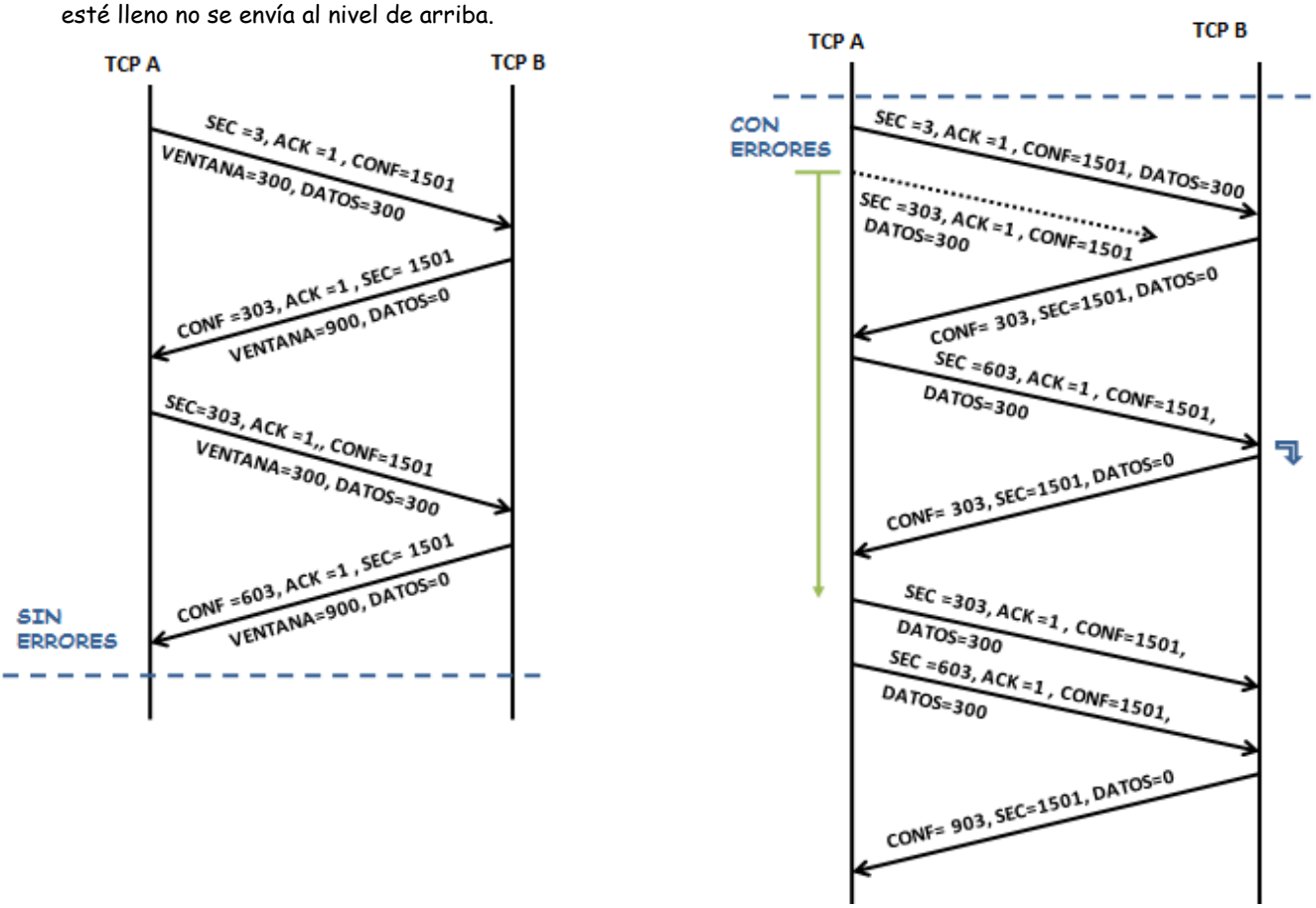
* **EJEMPLO:**



A desea un tamaño máximo de ventana de recepción de 300 octetos, B desea un tamaño máximo de ventana de recepción de 900 octetos.
Se indica en número de secuencia que desea usar cada una de las entidades, SEC=2 para la entidad A y SEC=1500 para B.
Una vez confirmados los segmentos de establecimiento de conexión de ambas entidades, el primer octeto de datos enviado por A se numerará a partir de 3 y en B a partir del 1501.

- **TRANSFERENCIA DE DATOS:**

El intercambio de octetos entre aplicaciones se realiza mediante unos buffers de envío y recepción. Estos buffers permiten el almacenaje de los datos enviados o transmitidos. Hasta que el buffer de recepción no esté lleno no se envía al nivel de arriba.



En este caso nos vamos a basar en una transmisión unidireccional de datos independientemente que durante la transmisión se produzcan errores o no.
La ventana de recepción en A es de 300 octetos y en B de 900.

-SIN ERRORES:

1. Se envía un segmento desde A hacia B de información que contiene 300 octetos en el campo de datos (DATOS= 300) además recuerda a la entidad TCP B el tamaño de la ventana de recepción de A (300 octetos).
2. La entidad TCP B envía un segmento sin datos (DATOS =0) en el que se confirman todos los octetos recibidos (CONF= 303)
   El campo CONF contiene el siguiente numero de octeto que se espera recibir, por tanto A entiende que todos los octetos enviados anteriormente han llegado bien. Además se recuerda el número de secuencia que va a utilizar y el tamaño de su ventana (900 octetos).

Estos dos pasos se van a producir con cada segmento de datos que se transmitan.

-CON ERRORES: Suponemos que la entidad TCP A va a transmitir 900 octetos, agrupados en tres segmentos pendientes de confirmación.

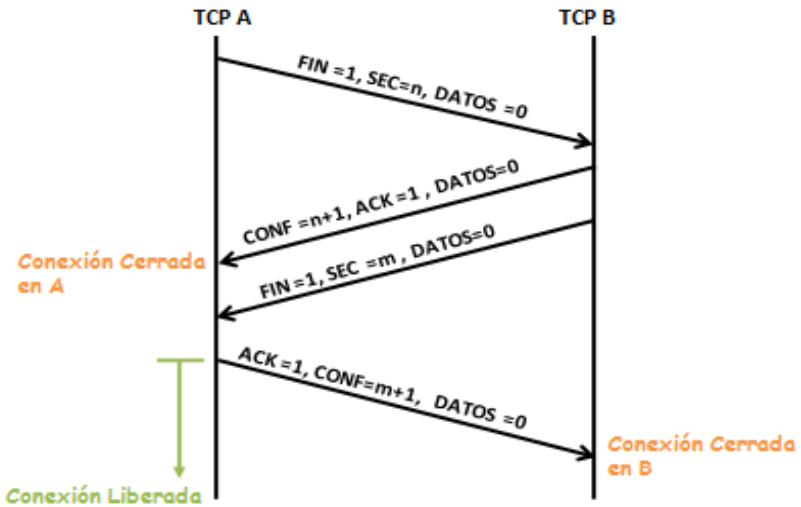
Además vamos a suponer que el segundo segmento se va a perder por el camino, al no haber retransmisión se va a tener que esperar el vencimiento del temporizador.

El tercer se va a enviar y recibir correctamente, pero como ha llegado fuera de secuencia se va a descartar. Desde el momento que se pierde un segmento la entidad B va a estar enviando su segmento solicitando la transmisión del segmento perdido.

Una vez que vence el temporizador se van a retransmitir los segmentos dos y tres y la estación B va a confirmar únicamente el ultimo (CONF=903), esto hace que también se confirme el segmento dos.

- **LIBERACION DE CONEXION:** Se realiza de forma ordenada.

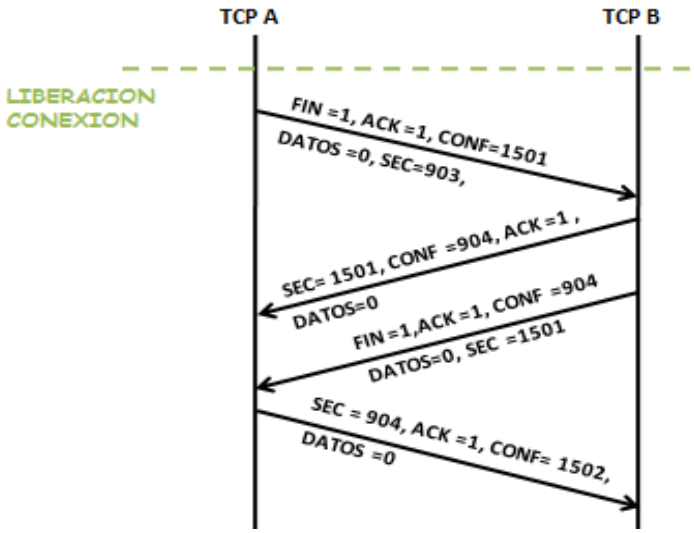
  Para liberar cada lado de la conexión, es necesario haber recibido previamente una confirmación de todos los octetos enviados. Esta liberación ordenada implica un cierre independiente en cada dirección de la conexión y cualquiera de las partes puede solicitar la liberación de la conexión.



Las dos partes pueden iniciar la liberación simultáneamente, en este caso, la liberación se completa cuando una de las partes ha enviado una confirmación. Como no hay confirmaciones de las confirmaciones, al transmitirse la última confirmación se lanza un temporizador con un tiempo estimado de llegada de esa confirmación.

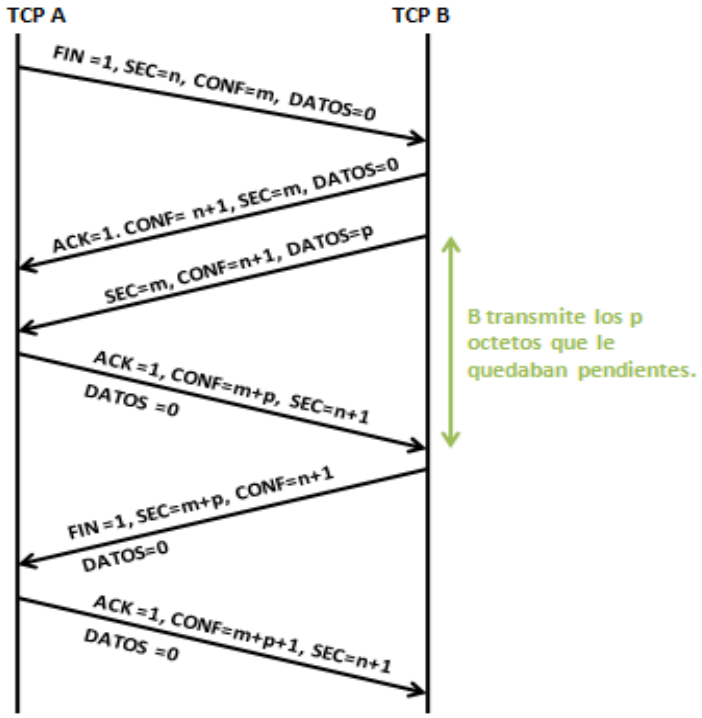
Cuando vence el temporizador se va a liberar oficialmente la conexión, este temporizador permite que dar un tiempo para que llegue la confirmación a su destino y a recibir potenciales segmentos retrasados u obsoletos para su eliminación inmediata.

* **EJEMPLO:** En este ejemplo vamos a considerar que no quedan datos pendientes por transmitir en ninguna de las dos estaciones.



La conexión se libera completamente cuando se transmite en cada sentido un segmento con el bit FIN activado.

-LIBERACION DE CONEXIÓN CON DATOS: Esta situación se da cuando una de las estaciones cierra la conexión por que ha terminado de transmitir datos y la otra continuar abierta hasta que termine de transmitir los datos que le quedan por enviar.



La estación B no va a cerrar la conexión hasta que primero no haya enviado todos los datos que tenga pendiente. Una vez que ya no quedan más datos enviara una solicitud de liberación (FIN =1).

1.5 Tipos y Tratamiento de Errores.

Los errores en la transmisión de segmentos TCP se pueden producir por dos motivos:

- **ERRORES FISICOS DE TRANSMISION:** Se producen en redes inalámbricas o radio enlaces móviles que pueden provocar pérdidas de paquetes a ráfagas.
  Altas tasas de pérdidas de bits causadas por las pobres condiciones de propagación.
- **SEGMENTOS PERDIDOS (DATOS Y/O ACKS):** En redes fijas (o de cable: Ethernet, líneas serie o anillos en fibra óptica) la mayoría de las perdidas es por congestión IP en los routers.
  En los sistemas finales se aplica un control de flujo para evitar pérdidas de datos extremo a extremo.

Mientras no se reciba correctamente el segmento de información perdido (o que haya llegado con errores de transición), se sigue indicando, repetidamente, que se continúa a la espera de sus octetos de datos ante la posterior llegada correcta de cualquier otro segmento de información no contiguo.

Si no se confirman los octetos de datos de los posteriores segmentos de información no contiguos que han llegado correctamente, se provoca el vencimiento de sus temporizadores en el lado emisor y su retransmisión.

Para evitar retransmisiones innecesarias se introduce la opción de confirmación selectiva (SACK: Selective Acknowledgment) que confirma los octetos de datos de segmentos de información no contiguos que han sido recibidos correctamente.

- **TIPOS DE CONFIRMACIONES (ACK'S):**
  - **-CONFIRMACION (NO DUPLICADA):** Se realiza la desactivación del temporizador del segmento de información que ha provocado dicha confirmación.
    Se elimina en el buffer de transmisión de la copia de los octetos de datos confirmados.
  - **-CONFIRMACION DUPLICADA:** No produce ningún efecto si se ha hecho antes, es decir, no se ha perdido el anterior ACK duplicado.
    El peligro está en que venza el temporizador del segmento de información no contiguo que ha provocado dicha confirmación repetida y se realiza una retransmisión innecesaria.
    Cuando al destino llegan segmentos TCP fuera de orden, el receptor transmite ACK'S duplicados o repetidos reconociendo que se han recibido segmentos no contiguos o fuera de orden.

-**CONFIRMACION DUPLICADA CON SACK:** Se realiza la desactivación del temporizador del segmento de información no contiguo. Se elimina en el buffer de transmisión de la copia de los octetos de datos no contiguos confirmados.

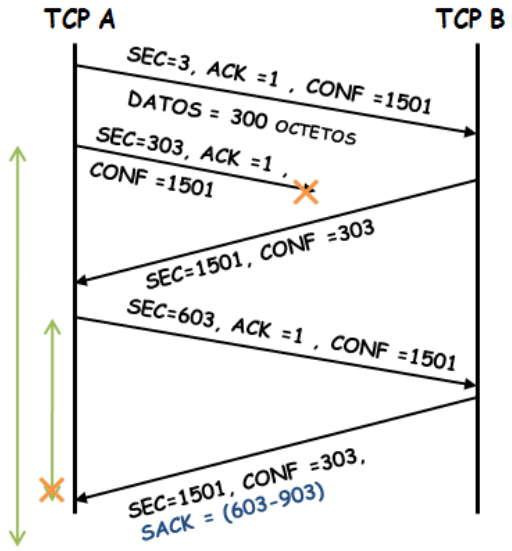
.

1.6 TCP SACK.

Propuesto para resolver el problema cuando hay múltiples perdidas de segmentos TCP. Evita retransmisiones innecesarias de segmentos TCP que ya fueron entregados al receptor.
Para poder hacer uso de la **OPCIÓN TIPO 5 SACK** en **FASE DE TRANSFERENCIA** de datos hay que indicarlo previamente en fase de establecimiento de la conexión TCP. En la **FASE DE ESTABLECIMIENTO** se indica mediante la **OPCIÓN TIPO4 TCP SACK PERMITTED** en un segmento SYN =1.

Mediante la opción TIPO 5 SACK-PERMITTED se informa al emisor de bloques no contiguos de octetos de datos que han sido recibidos correctamente y, por tanto, en donde puede haber "agujeros en los datos". Esta opción provoca la cancelación en el emisor de los temporizadores de los segmentos que sabe que han llegado correctamente al receptor y la eliminación de la copia de los correspondientes octetos de datos en el buffer de transmisión.
Si el receptor no ha recibido la opción TIPO 4 TCP SACK-PERMITTED en el establecimiento de la conexión no debe usar la opción TIPO 5 TCP SACK-PERMITTED en la fase de transferencia de datos.

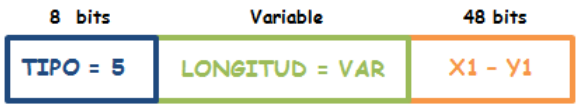
* **EJEMPLO:**



A va a transmitir 900 octetos agrupados en 3 segmentos pendientes de confirmación, suponiendo que el segundo segmento se pierde.

B emplea SACK por tanto cuando llega un segmento fuera de secuencia se va a seguir pidiendo el segmento que falta pero gracias a la opción SACK TCP A sabe que los octetos del 303al 602 no han llegado todavía pero sí los octetos del 603 al 902 y va a cancelar el temporizador que se lanza cuando se envía el SEC=603.

- **PAQUETE SACK CON 1 BLOQUE PARA SEGMENTOS NO CONTIGUOS Y CONSECUTIVOS:**

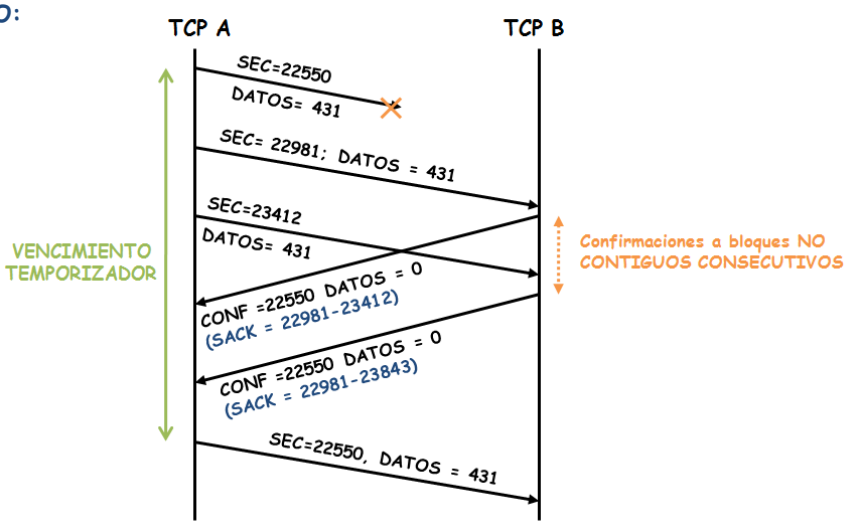


-**X1 (32 bits): BORDE IZQUIERDO DEL BLOQUE "1".** Indica el primer octeto de datos recibido del "primer" segmento de información no contiguo que ha hecho enviar el primer paquete SACK.
X1 es el mismo valor para todos los paquetes SACK siempre que los siguientes segmentos de información no contiguos recibidos sean consecutivos.

-Y1 (32 bits): BORDE DERECHO DEL BLOQUE "1". Indica el primer octeto de datos que se espera recibir del siguiente segmento de información contiguo y consecutivo.

Si los octetos de datos recibidos del siguiente segmento de información no contiguo son consecutivos, entonces, se agrupan éstos en el mismo bloque X1-Y1.

* EJEMPLO:



Se intenta agrupar el máximo de octetos da datos consecutivos en cada paquete SACK.

Ante la llegada de cualquier segmento de información no contiguo y consecutivo, se indican siempre los octetos de datos ya recibidos y se agrupan los octetos del último segmento en el mismo bloque.

- PAQUETE SACK CON N BLOQUES PARA SEGMENTOS NO CONTIGUOS Y NO CONSECUTIVOS:

Si empleamos casi todo el espacio del campo de opciones TCP (máximo de 40 octetos) podemos enviar hasta **4 BLOQUES NO CONSECUTIVOS SACK (34 octetos).**

Si los nuevos octetos de datos son consecutivos con los anteriores ya recibidos, entonces, se agrupan en un mismo primer bloque X1-Y1 y no entrarían dentro de los 4 bloques permitidos.

EL BLOQUE 1 (X1-Y1) DEBE INFORMAR DEL SEGMENTO RECIBIDO MAS RECIENTEMENTE. Asegura que el ACK con la opción SACK refleje el cambio más reciente en el buffer de recepción. Así el emisor tiene información actualizada del estado de la red y del estado de la cola de recepción.

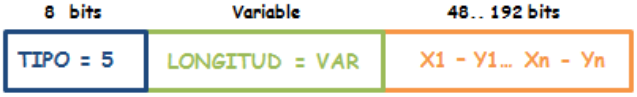
Si llega un nuevo segmento y se puede agrupar sus octetos con otro y otros bloques ya recibidos, entonces, se agrupan todos los octetos que se puedan en un primer bloque.

* EJEMPLO:

Suponemos que llega datos = (6500-7000)

(6500-7000) (8000-8500) (7000-7500) (6000-6500) ==> (6000-7500) (8000-8500)

Después del primer bloque SACK, los siguientes pueden estar listados en orden arbitrario.



-X1 (32 bits): BORDE IZQUIERDO DEL BLOQUE "1". Indica el primer octeto de datos recibido del segmento de información no contiguo que ha hecho enviar el paquete SACK.

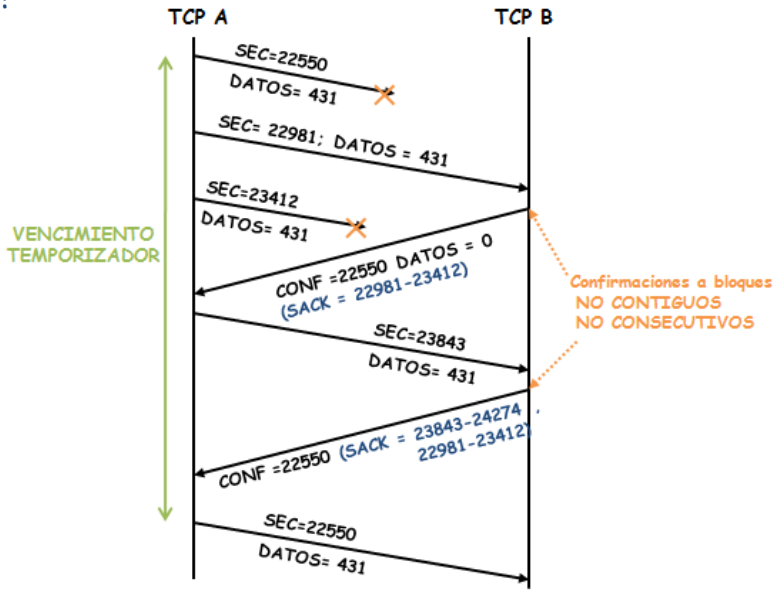
X1 es el mismo valor para todos los paquetes SACK siempre que los siguientes segmentos de información no contiguos recibidos sean consecutivos.

-Y1 (32 bits): BORDE DERECHO DEL BLOQUE "1". Indica el primer octeto de datos (siguiente al último octeto del bloque recibido) que se espera recibir del siguiente segmento de información contiguo y consecutivo.

-Xn (32 bits): BORDE IZQUIERDO DEL BLOQUE "n" NO CONSECUTIVO. Recordatorio del primer octeto de datos ya recibido de un segmento de información no contiguo y no consecutivo.

-Yn (32 bits): BORDE DERECHO DEL BLOQUE "n" NO CONSECUTIVO. Recordatorio del primer octeto de datos que se espera recibir del siguiente segmento de información no contiguo y consecutivo.

* EJEMPLO:



1.7 CONTROL DE CONGESTION.

El mecanismo de control de flujo de TCP se diseño para permitir que el destino restrinja el flujo de segmentos de una fuente y evitar así la saturación de la memoria temporal del destino. Este mismo mecanismo se utiliza ahora para proporcionar control de congestión sobre internet entre la fuente y el destino.
La congestión tiene dos efectos principales:

1. Cuando la congestione pieza a producirse, el tiempo de transmisión a través de la red aumenta.
2. Conforme la congestión se hace más severa, la red empieza a descartar paquetes.

El mecanismo de control de flujo de TCP se puede utilizar para identificar el comienzo de la congestión y reaccionar mediante la reducción del flujo de datos. Si muchas entidades TCP operan de esta manera la congestión se puede aliviar.

- GESTION DE TEMPORIZADORES EN TCP:

  No es lo mismo que el destino este ubicado en la misma Ethernet que disperso geográficamente en Internet. La elección de valores adecuados es mucho más compleja en el nivel de transporte que en el nivel de enlace, esto se debe a:

  1. Diferencia en capacidad y retardo de unas conexiones TCP a otras.
  2. Oscilaciones debidas a la presencia de routers y situaciones de congestión que están fuera de sus control
  3. Aun en situaciones de control, los routers pueden tener largas colas de datagramas IP que atender, los enlaces pueden ser de diferentes velocidades y retardos y la ruta puede variar durante la conexión.

  La elección de un valor adecuado tiene una consecuencia directa en el funcionamiento eficiente de TCP. Si un temporizador es demasiado alto, el emisor esperara innecesariamente en ciertos casos por ACK'S que nunca llegaran. Si el temporizador es demasiado bajo se producirán retransmisiones innecesarias de segmentos que habían sido correctamente recibidos.
  Manejar los **TEMPORIZADORES** es siempre complejo porque hay que determinar "aproximadamente" el periodo de tiempo existente desde que se envía un segmento hasta que se recibe su ACK, es decir, **EL TIEMPO DE IDA Y EL TIEMPO VUELTA (RTT)**

Los valores del temporizador se establecen mediante ALGORITMOS AUTOADAPTATIVOS que dinámicamente ajustan los valores al estado de la red, según es percibido éste por la entidad de transporte emisora.

-ALGORITMO AUTOADAPTATIVO DE KARN:

Se usa para el cálculo del RTT: Se toma una muestra RTT de cada segmento de información transmitido. Se obtiene un promedio de dichas muestras y se le añade un margen de seguridad.

- **CONGESTIÓN EN INTERNET:**

La congestión implica la pérdida de 1 o más datagramas, las posibles causas para que se produzca congestión en Internet son, que los enlaces de salida sean de baja capacidad y el exceso de tráfico de entrada en alguna o algunas redes de Internet, es decir, tasas de entrada que superan las capacidades de salida.

En principio no se pueden conocer todos los sistemas intermedios. Cuando los datagramas llegan a la parte congestionada de la red, como el router de acceso no puede aceptar todo el tráfico entrante, empezara a acumularlos en su buffer y cuando éste se llene, empezará a descartar.

Basta que uno de los enlace del trayecto se vea afectado por la congestión para limitar el tráfico en todo el camino y, por tanto, el rendimiento de la comunicación entre los sistemas finales. Las aplicaciones se bloquean o dejan de funcionar.

La entidad TCP receptora recibirá solo una parte de los segmentos por lo que la entidad emisora retransmitirá por Timeout aquellos que no hayan sido confirmados.

La VENTANA DE RECEPCIÓN garantiza que no se inunde al receptor ejerciendo un control de flujo sobre el emisor. Sin embargo los problemas podrían estar en Internet, muchas veces el cuello de botella está en la red y no en el receptor. Los buffer de los routers pueden desbordarse. Por tanto, además de control de flujo necesitamos control de congestión.

La ventana de recepción se notifica al emisor por el receptor: el control de flujo lo ejerce la entidad TCP receptora.

-VENTANA DE CONGESTION:

Para notificar a la entidad TCP emisora que hay una saturación en algún punto del trayecto y de que debe bajar el ritmo de transmisión se utiliza una VENTANA DE CONGESTION que le indique la cantidad de octetos de datos que puede enviar en un momento dado.

LA VENTANA DE CONGESTION ACTUA SIMULTÁNEAMENTE Y PARELELO CON LA VENTANA DE RECEPCIÓN.

En cada momento, el emisor tomara en consideración la más pequeña de las dos ventanas para asegurarse de que no satura el receptor ni provoca, o contribuye a agravar una situación de congestión en la red.

La ventana de congestión la calcula el emisor a partir de la cantidad de retransmisiones (perdidas de segmentos o timeouts) que tiene que realizar: EL CONTROL DE LA CONGESTIÓN LO EJERCE LA ENTIDAD TCP EMISORA.

El emisor emplea un mecanismo indirecto basado en la perdida de datagramas IP por la red.

El emisor, en principio, considera la red altamente fiable hasta que no reciba los ACK correspondientes (segmentos TCP que no han llegado al destino), por lo que deduce que la red esta descartando datagramas IP por congestión y reducirá el ritmo de sus envíos.

El emisor al comprobar que no se producen retransmisiones va aumentando paulatinamente la ventana hasta llegar al punto donde falla algún segmento, momento en el cual reduce, siempre y cuando la ventana de recepción no imponga ninguna limitación.

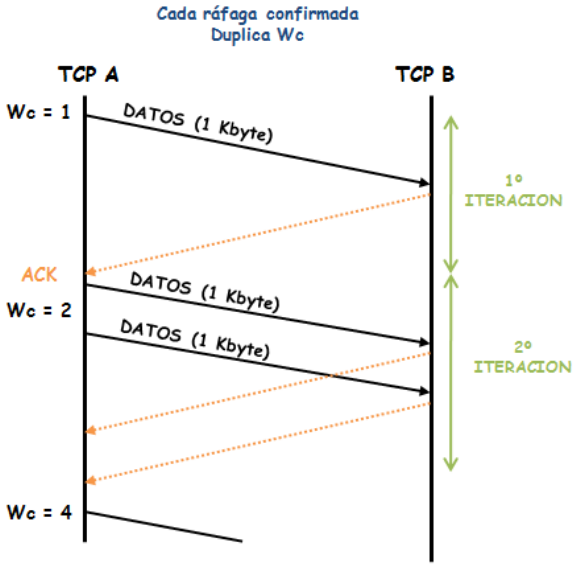
La VENTANA DE CONGESTION, mediante un algoritmo adaptativo, crece de forma lenta y gradual mientras que la reducción se lleva a cabo de manera drástica.

- **MECANISMOS DE CONTROL DE LA CONGESTIÓN:**

-COMIENZO LENTO o SS(SLOW START):

* **OBJETIVO:** Tantear la red, transmitiendo tan rápido como sea posible, hasta que se produzca una congestión

Con SS se transmiten segmentos de información de forma exponencial (1, 2, 4, 8, …) al ritmo al que se reciben los ACK desde el otro extremo.



Wc, que se mide en segmentos (1 segmento = 1MSS de 1 Kbyte), crece exponencialmente sólo si todos los segmentos de información enviados han sido confirmados. Si hay un ACK retrasado, el incremento en el tamaño de Wc no es exponencial.

Se denomina **COMIENZO LENTO** porque se empieza a transmitir lentamente, primero 1 segmento, luego 2 segmentos, y así sucesivamente; pero en realidad se transmite muy rápido.

En solo 7 interacciones (envío de un grupo de 64 segmentos de información) se llega a 65.535 octetos que es el valor máximo de TCP para W_R y que coincide con el **VALOR INICIAL DEL UMBRAL DE CONGESTION O UMBRAL DE PELIGRO TCP,** el cual va variando en función de la congestión en Internet.

* **FUNCIONAMIENTO:**

1. Wc =1 : inicialmente 1 segmento o 1 MSS del tamaño máximo para la conexión.
2. Wc = 1, 2, 4, 8 .. (Wc= W_T) va creciendo exponencialmente hasta superar:
   - La W_R actual y puntual (control flujo)
   - El umbral de congestión o umbral de peligro (limite de crecimiento) máximo impuesto por TCP inicialmente para W_R de 65.535 octetos.
   - El buffer de algún router y se produzca la primera perdida.
3. Cada vez que se detecta una pérdida se obtiene un nuevo umbral de congestión o peligro, que toma como valor la mitad del tamaño actual de Wc.

   A su vez, Wc se reduce de una manera drástica al valor inicial de 1 segmento. Ahora Wc empieza a crecer exponencialmente como antes, pero solo hasta el nuevo umbral de congestión.
   - **NUEVO UMBRAL = Wc(actual)/2**
   - **Wc = 1:** Reducción rápida y significativa del tráfico para que el routers se recupere. Mientras el tamaño máximo de Wc permanezca en el umbral de congestión, no se enviara una ráfaga de mayor longitud, sin importar la cantidad de espacio de W_R ofrecida por el receptor en el supuesto de ser superior.

-PREVENCION DE LA CONGESTION o CA (CONGESTION AVOIDANCE):

* **OBJETIVO:** Prevenir la cogestión. Una vez que se ha producido la congestión, procura evitar que se reproduzca en la red mediante una transmisión mucho más lenta.

Con CA si la red se congestiona es deseable que esta se recupere sin dejar de transmitir. Una vez que se ha producido la congestión, CA, procura evitar que se reproduzca mediante una transmisión mucho más lenta.

Al producirse una congestión (detección de perdida), se calcula el nuevo umbral (Wc(actual)/2) y ahora, se incrementa Wc linealmente desde el nuevo umbral hasta que se supere el tamaño máximo de W_R o el buffer de algún router.

Cada vez que se detecte una pérdida de segmento, se calcula el nuevo umbral y se continúa transmitiendo linealmente a partir de dicho umbral.

CA en combinación con SS permite seguir transmitiendo a partir de donde SS finaliza.

* **FUNCIONAMIENTO:** Partimos del punto 3 del SS

4. A partir del nuevo umbral, se aplica CA y se incrementa Wc linealmente de 1 en 1 hasta:.
   - **UNA NUEVA PERDIDA (CONGESTION).** Momento a partir del cual se aplica de nuevo SS, obteniéndose un nuevo umbral de congestión, en un valor igual a la mitad del tamaño actual de Wc

     A su vez, Wc se reduce de una manera drástica al valor inicial de 1 segmento. Ahora Wc empieza a crecer exponencialmente como antes, pero solo hasta el nuevo umbral de congestión.
   - **ALCANZAR TAMAÑO DE W_R:** Momento a partir del cual se aplica un control de flujo. En este punto Wc dejara de crecer y permanecerá constante mientras no ocurran mas pérdidas y W_R no cambie de tamaño.

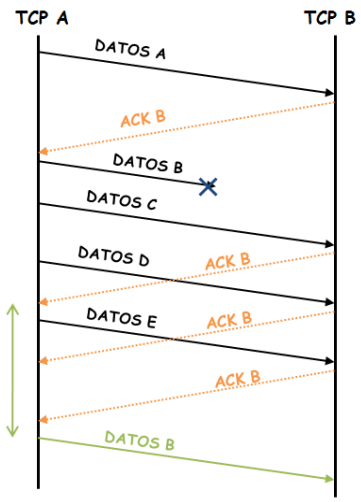
-RETRANSMISION RAPIDA (FAST RETRANSMIT):

Mientras no se reciba correctamente el segmento perdido (o con errores), se sigue confirmando, repetidamente, que se espera la llegada correcta de sus octetos de datos, a la llegada de cualquier otro segmento de datos.

Recepción correcta de segmento fuera de secuencia, conlleva la generación inmediata de una confirmación (ACK) duplicada o repetida reconociendo que se ha recibido un segmento de información fuera de secuencia.

Las causas de que un segmento llegue fuera de secuencia son, que los segmentos llegan al receptor desordenados, que se han perdido segmentos y que los segmentos llegan con errores de transmisión.

* **RETRANSMISION TCP:** Se va a producir una retransmisión de un segmento TCP si:
  - **EXPIRA EL TEMPORIZADOR DE RETRANSMISION:** Perdida segura del segmento, hay una alta seguridad de que haya congestión en Internet.
  - **LLEGAN 3 CONFIMACIONES (ACK'S) DUPLICADOS:** Perdida probable del segmento, no hay seguridad de que haya congestión en Internet, pero por si acaso, no se espera al vencimiento del temporizador y se aplica el mecanismo de retransmisión rápida.
* **EJEMPLO:** La retransmisión del segmento se va a realizar después de la recepción de 3 ACK duplicados sin esperar el timeout.



Cuando se reciben 3 ACK duplicados, o sea, 4 ACK idénticos en secuencia, significa que se ha perdido un segmento.

La RETRANSMISION RAPIDA no es mecanismo estricto de control de congestión en Internet. Es un mecanismo de retransmisión de la forma más rápida posible, no espera el vencimiento del temporizador, de un segmento de información y que, posiblemente se ha perdido, pero no es seguro, debido a una presumible congestión en un router.
En combinación con el mecanismo CA da origen a un nuevo mecanismo de control de la congestión denominado Recuperación Rápida.

Cuando la detección se debe a un vencimiento de un temporizador, comienza de nuevo la fase SS y después, a partir del umbral, la fase de CA. Seguro que se ha perdido el segmento y hay congestión.
Cuando la detección se debe a 3 ACK'S se comienza directamente una nueva fase CA sin comenzar a crecer exponencialmente hasta el umbral (SS) y, luego, linealmente (CA) = MECANISMO DE CONTROL DE LA CONGESTION DE RECUPERACIÓN RÁPIDA. del segmento, hay una alta seguridad de que haya congestión en Internet.

-RECUPERACION RAPIDA (FAST RECOVERY):

Si la detección se debe a 3 ACK duplicados se hace una retransmisión rápida y se comienza una nueva fase de CA a partir del nuevo umbral sin comenzar en SS.

* **EJEMPLO:** Si se han recibido 3 ACK y el nuevo umbral= 6, se comienza linealmente a partir de 7, 8, 9 …, porque tal vez no hay congestión.

- **NOTIFICACIONES DE CONGESTION EN TCP:**

-NOTIFICACIÓN O SEÑALIZACION IMPLICITA POR: Vencimiento de temporizador (timeout) o por la recepción de 3 ACKS duplicados.

-NOTIFICACIÓN O SEÑALIZACION EXPLICITA POR:

* **ECN (Explicit Congestion Notification):** La transmisión se ajusta en la entidad TCP emisora sólo cuando una entidad IP intermedia se lo notifica expresamente a la entidad TCP receptora y esta se lo comunique, finalmente a la entidad TCP emisora.
* **MENSAJE ICMPv4 DE FRENADO EN EL ORIGEN (TIPO=4, CODIGO=0):** No se usa por seguridad (prevención de que ataque aun router). Además no existe el mensaje ICMPv6 de frenado de router.

- **ECN:**

-FUNCIONAMIENTO:

El EMISOR ACTIVA el bit de Transporte de Notificación Explicita de Congestión o ECT DE LA CABECERA IP solicitando Notificación Explicita de Congestión (ECN) para este datagrama a cualquier router que detecte congestión.
El ROUTER al detectar el bit ECT=1 y congestión, ACTIVA EL BIT de Detección de Congestión o EC de la CABECERA IP y lo encamina a la entidad TCP receptora.
La ENTIDAD RECEPTORA TCP, al recibir dicha cabecera IP, ACTIVA EL BIT de eco de Notificación Explicita de Congestión (ECN-ECHO) DE LA CABECERA TCP y se lo envía a la entidad TCP emisora para que esta active el control de la congestión.
La ENTIDAD EMISORA TCP ACTIVA el bit de Ventana de Congestión Reducida o CWR de CABECERA TCP y se lo envía a la entidad TCP receptora para indicar que ya ha activado el control de congestión y que no le notifique la congestión otra vez.

-CABECERA TCP-ECN:

Se utilizan dos bits en la cabecera TCP para negociar la funcionalidad ECN: ECN-ECHO y CWR.

El transmisor activa dichos bits en el segmento SYN de establecimiento y el receptos actica el bit ECN.ECHO en el segmento SYN ACK.

* **ECN-ECHO:** El receptor continúa activando el bit ECN-ECHO hasta recibir un segmento con el bit CWR
* **CWR:** El transmisor activa el bit CWR para indicar que ya ha actualizado el valor de la ventana de congestión.

- **PROBLEMÁTICA DEL CONTROL DE CONGESTION TCP:**

Solo está pensado para aplicaciones cobre TCP y cada vez hay más aplicación sobre UDP, como es el trafico en tiempo real, VoIP y aplicaciones de audio y video.

Los usuarios tienden a desactivar los controles de congestión TCP.

Las retransmisiones TCP son inaceptables para aplicaciones multimedia interactivas en tiempo real al incrementar el retardo extremo a extremo. Además el control de congestión TCP reduce la tasa de envío al emisor.

Surge la necesidad de implementar mecanismos de congestión a nivel de red.

2. PROTOCOLO UDP

- **CARACTERISTICAS:**

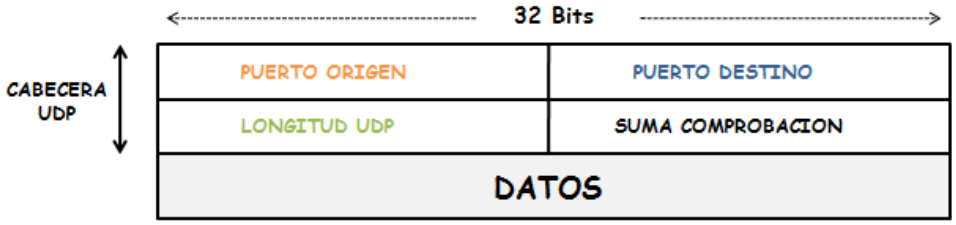
1. No ofrece fiabilidad extremo a extremo.
2. Ofrece un servicio no orientado a conexión. No ofrece control sobre errores lógicos (detección y recuperación), no control de flujo.
3. Ofrece detección opcional, sin recuperación, de errores físicos. Opcionalmente se comprueba la integridad de la cabecera y datos del datagrama UDP, utilizando un método similar al de suma comprobación.
4. Multiplexación (Origen) y Demultiplexación (destino) en función de los números de puerto.
5. Transferencias en modo full_duplex.

- **DATAGRAMA UDP:**

-DATOS: Incluye la cabecera de información del nivel de aplicación y los datos del mensaje (si existen).

Los datos deben pasarse en bloques bien delimitados ya que no existe un servicio de flujo de octetos como en TCP.

-CABECERA: Tiene longitud mínima de 8 octetos.



* **PUERTO ORIGEN/DESTINO:** Limitados a 16 bits cada uno. Identifican al proceso de aplicación emisor / receptor que envía un segmento UDP.
* **LONGITUD UDP:** Indica la longitud en octetos del datagrama UDP completo (cabecera y datos).
* **SUMA COMPROBACION:** Suma aritmética binaria o en modulo 2 de todos los bloques de 16 bits del datagrama. Si una entidad UDP no desea calcular la suma el campo debe ser cero.

Cuando en un datagrama se detecta un error se elimina y no se entrega al proceso de aplicación